Amendments to the Claims:

This listing of claims will replace all prior versions and listings of claims in the application:

1. (currently amended) A method for sharing a class among a plurality of applications in a multitasking computer system, the method comprising:

~~extracting~~ removing one or more static fields from the class;

creating a separate copy of the one or more static fields for each of the plurality of applications that utilizes the class, wherein each of the separate copies corresponds to one of the plurality of applications;

creating one or more access methods for the one or more static fields, wherein the access methods are operable to access the corresponding separate copy of the one or more static fields based upon the identity of the utilizing application; and

initializing each separate copy of the static fields once, wherein the initializing includes:

embedding in a class constructor one or more instructions for performing the initializing each separate copy of the static fields; and

executing the class constructor once for each separate copy of the static fields.

2. (canceled)

3. (original) The method of claim 1, further comprising:

loading a template class for each separate copy of the static fields, wherein the template class comprises a static initializer for one of the separate copies of the static fields; and

executing the static initializer once for each separate copy of the static fields.

4. (original) The method of claim 3, further comprising:

renaming the template class with a substantially unique name for each separate copy of the static fields.

5. (original) The method of claim 4, further comprising:
storing the renamed template class on a storage medium.

6. (original) The method of claim 3, further comprising:
associating a class loader with each of the plurality of applications, wherein each class loader is executable to perform the loading the template class for each separate copy of the static fields.

7. (original) The method of claim 1,
wherein the multitasking computer system comprises a virtual machine, and wherein the applications are executable by the virtual machine.

8. (original) The method of claim 1,
wherein the plurality of applications are executable in a platform-independent programming environment.

9. (currently amended) A carrier medium comprising program instructions for sharing a class among a plurality of applications in a multitasking computer system, wherein the program instructions are computer-executable to implement:
~~extracting~~ removing one or more static fields from the class;
creating a separate copy of the one or more static fields for each of the plurality of applications that utilizes the class, wherein each of the separate copies corresponds to one of the plurality of applications;
creating one or more access methods for the one or more static fields, wherein the access methods are operable to access the corresponding separate copy of the one or more static fields based upon the identity of the utilizing application; and

initializing each separate copy of the static fields once, wherein the initializing includes:

embedding in a class constructor one or more instructions for performing the initializing each separate copy of the static fields; and

executing the class constructor once for each separate copy of the static fields.

10. (canceled)

11. (original) The carrier medium of claim 9, wherein the program instructions are further computer-executable to implement:

loading a template class for each separate copy of the static fields, wherein the template class comprises a static initializer for one of the separate copies of the static fields; and

executing the static initializer once for each separate copy of the static fields.

12. (original) The carrier medium of claim 11, wherein the program instructions are further computer-executable to implement:

renaming the template class with a substantially unique name for each separate copy of the static fields.

13. (original) The carrier medium of claim 12, wherein the program instructions are further computer-executable to implement:

storing the renamed template class on a storage medium.

14. (original) The carrier medium of claim 11, wherein the program instructions are further computer-executable to implement:

associating a class loader with each of the plurality of applications, wherein each class loader is executable to perform the loading the template class for each separate copy of the static fields.

15. (original) The carrier medium of claim 9,

wherein the multitasking computer system comprises a virtual machine, and wherein the applications are executable by the virtual machine.

16. (original) The carrier medium of claim 9,

wherein the plurality of applications are executable in a platform-independent programming environment.

17. (currently amended) A multitasking computer system for isolating the execution of a plurality of applications, the system comprising:

an original class utilized by the plurality of applications, wherein the original class comprises one or more static fields;

a modified original class generated from the original class, wherein the modified class comprises the original class without the static fields; [[and]]

a static field class generated from the original class, wherein each instance of the static field class corresponds to one of the plurality of applications, wherein the static field class comprises one or more instance fields corresponding to the one or more static fields of the original class, and wherein each instance of the static field class is initialized once; and

a class constructor which comprises one or more instructions for initializing each instance of the static field class, and wherein the class constructor is configured to be executed once for each instance of the static field class.

18. (canceled)

19. (original) The system of claim 17, further comprising:

a template class, wherein the template class is loaded for each instance of the static field class, wherein the template class comprises a static initializer for one of the instances of the static field class, and wherein the static initializer is configured to be executed once for each instance of the static field class.

20. (original) The system of claim 19,

wherein the template class is configured to be renamed with a substantially unique name for each instance of the static field class.

21. (original) The system of claim 20, further comprising:

a storage device which is operable to store the renamed template class.

22. (original) The system of claim 19, further comprising:

a plurality of class loaders, wherein each of the plurality of applications is associated with one of the class loaders, and wherein each class loader is executable to load the template class for each instance of the static field class.

23. (original) The system of claim 17, further comprising:

a virtual machine which is configured to execute the plurality of applications.

24. (original) The system of claim 17, further comprising:

a platform-independent programming environment in which the plurality of applications are executable.